The user specifies if particular functions should be executed prior to or after the filters are executed. If the user wants, for example, to not display tax order lines but wants to include the tax amount from those lines in the total order amount, the function to calculate total order amount from the order lines occurs prior to filtering out the tax order lines. If the user wants, for example, to not display tax order lines and does want to include the tax amount from those lines in the total order amount, the function is executed after filtering out the tax order lines.

The present invention attempts to minimize the time it takes to run the functions for a new set of management record instances (MRIs) by running each function only once and by reading each PRI as few times as necessary. Because functions can include the results (calculated fields) of other functions, the present invention "restates" functions to improve calculation speed. The present invention first ensures that all functions are restated as only PRT fields and group by functions of PRT fields. Restatement eliminates (except for group by functions) one function being dependent upon another function because that function uses the result of the other function. Restatement occurs for each function when the user first defines (enters or maintains) it or one of its component functions. Besides replacing calculated fields with the fields from which they are calculated, the present invention also determines in which pass a specific function should be processed. For a particular MRI, the present invention must make multiple passes when a group by field from one level in the MRPF hierarchy is used in a function in another level in the MRPF hierarchy and the result of that function is used in another function in the original group by function hierarchy level. For each function, the present invention checks to see if that function is comprised of any function which uses a group by fields from a different hierarchy or uses a calculated field from a hierarchy level higher than the current level. If either of the above two conditions exist, the present invention adds one to the calculation pass sequence number. The present invention then runs the same routine on the subfunctions that meet the test conditions. The present invention continues this process until it has no more sub-functions to test. As a result of this process, each function has a pass sequence number.

As shown in FIGS. 7A and 7B, the method starts with step 150 in which the MRT definitions are read. The present invention processes each MRI separately. Then, in step 151, a MRI is read and the pass sequence number is set equal to zero. In step 152, the method reads the lowest

unprocessed MRPF level for this pass. First, the method executes for each MRPI all the functions with a pass sequence number of zero at the same time. It starts with the MRPFs that are lowest in the hierarchy and works its way up. In step 153, the method reads the MRPF's functions. Then in step 154, the method reads other MRPF's group by functions that use this level. Group by functions are executed with the MRPF they summarize, not the MRPF where the result is stored. The MRPIs of the MRPF are then read in step 155. Then in step 156, the functions for this MRPF are executed and added to the MRPI. The group by accumulators is then updated. Group by accumulators store not only the results of the group by calculation, but also enough information to recalculate the amount without rereading all the MRPIs if one of the component MRPI changes. For example, a "mean" group by accumulator includes not only the means but also the total number of MRPIs. Next in step 158, the system determines whether the MRPI being used to execute the functions is the last MRPI for this MRPF. If not, the method returns to step 155, otherwise the process continues in step 159. In step 159, the group by results is added to the parent MRPI. Then in step 160, the method test whether this is the last MRPF for this MRT. If not, the process loops to step 152 to process any unprocessed MRPFs, otherwise the process continues in step 161. In step 161, the method determines if this is the last pass. The higher the pass sequence number that a group by function has the latter it gets processed. The sequence number is based on the nesting of the group by function. If it is not the last pass, then 1 is added to the pass sequence number in step 162 and the method returns to step 152 for further processing. After each pass of all the MRPIs, the method executes the functions with the next highest pass sequence number until all functions are executed. However, if it is the last pass, the method continues in step 163 where the process tests whether this is the last MRI to perform functions for. If not, the process returns to step 151, otherwise the execution of the functions for the MRT is complete.

The present invention also provides for manipulation of the data using filters. The filters of the present invention are advantageous because the user does not need to join the underlying tables together in his/her select statement. Users can select a subset of all possible MRTs and a subset of the MRPFs that comprise the selected MRTs. Section criteria are applied against any PRT or calculated field in the MRT. Where there could be multiple occurrences of the field in a MRT (such as quantities ordered on one customer order), the user must also specify a group by filter just as with functions. The present invention has the two special group by filters "any" and "all." The other

selection criteria are: equal to, not equal to, greater than, less than, like, and exists in dynamic document X. Filters can be comprised of multiple selection criteria. The criteria can be nested and include AND/OR conditions. An example of a filter is give me all MRTs where the customer is in California, and the total of the order line amounts is greater than $10,000, and any of the Salespersons are in territory one, or the order is on hold.

Any number of filters for a MRT or its MRPFs can be active at any time. The user defines as many filters as he/she wishes and then activates and deactivates them whenever he/she wants. Filters are executed either when the MRIs are fetched by the production data source or after they are stored in memory 18. The user specifies which filters should be part of the fetch. Because of performance concerns, fetches with group by functions cannot be used for fetches. The user can also indicate if the filter is for just temporarily hiding the data or for deleting the data. If a parent MRPI is filtered out, so are its children MRPIs. Filters for MRTs are handled exactly like filters on the top most MRPF in the hierarchy.

FIGS. 8A and 8B illustrate the preferred process for executing filters. The process begins by reading any new or changed MRPI in step 170. Then a filter from the MRT definition is read in step 171. Next, in step 172, an operand is tested, and the result is saved. Since all filters operate on fields in the MRPF or on fields which have been grouped by a function into the MRPF, each MRPI can be fully tested to see if it should be filtered out by reading only the MRPI's calculated fields and its PRT's PRI field values. For improved processing efficiency, the present invention executes all of a MRPI's filter tests one after the other. The operation tests are simple Boolean tests to see if the field's value equals the value stored in the filter, or stored in the specified variable, or is in the list (Dynamic Document) specified. The present invention stores the result (true or false) of each operation. The method then determines if this operand is the last operand to test in step 173. If not, the process returns to step 172 to test the next operand. If all the operands have been tested, then the process continues to step 174. In the step 174, the method looks at the results of each operand to determine if the entire filter results in a pass or fail. If the MRPI passed the filter, the method proceeds to step 177 where the method appends to the MRPI a flag for each filter indicating if the filter includes that particular MRPI. If the MRPI fails the filter, the method jumps to step 178 to evaluate the filter type. In step 178, further tests are performed to determine if the filter is a delete